Jupyter Notebook showing analysis of tweets for the chosen dataset

- The below shows the graphs used to analyse the data set.

Contents:

Structure

- Shows the Pie Chart of the different types of tweets such as tweets, retweets and replies
- Mean of the different types
- Standard deviation of the different types

- Timeline

- Shows the time line of the the tweets, retweets and replies during t he period of the data set

- Sources

- Shows the applications used to send tweets

- Hashtags

- Shows the hashtag cloud of the data set

- Interactions

- Network graph
- retweets, replies, mentions

The set of code below imports the necessary python libraries in order to produce the analysis of the data set

In [1]:

```
import pandas
from parse import parse_data
from analyse import analyse_types
from analyse import analyse_entities
from analyse import analyse_replies
from analyse import analyse_field
from analyse import analyse_retweets
from analyse import analyse_relations
from analyse import analyse_types_field
from analyse import analyse_interactions
from encode import encode_json
data = parse_data("../Data/digifest16.csv")
```

data.head(10) below shows the first 10 entries of the dateframe that is used in order to extract necessary information used to analyse the set.

In [2]:

```
data.head(10)
```

Out[2]:

	id_str	from_user	text	created_at	user_lang
0	705802747971543040	laurajisc	More power to you' - @Jisc's opening video fro...	2016-03-04 17:11:02	en
1	705802727276847104	mark8n	More power to you' - @Jisc's opening video fro...	2016-03-04 17:10:58	en
2	705800984661004289	JournalArchives	RT @Jisc: Timelapse action from #digifest16. S...	2016-03-04 17:04:02	en
3	705800366370250753	dr_mike_jones	Presentations, podcasts & resources from @...	2016-03-04 17:01:35	en
4	705800366328307712	JiscLondon	Presentations, podcasts & resources from @...	2016-03-04 17:01:35	en
5	705800203752882176	HistoricalTexts	RT @Jisc: Slides and resources from two very b...	2016-03-04 17:00:56	en
6	705800083930005504	wagjuer	RT @coolcatteacher: Get #Digifest16 slides and...	2016-03-04 17:00:27	en
7	705799329001422848	MobileTechExp	RT @ResourceProds: We have lots of digital/cre...	2016-03-04 16:57:27	en
8	705798453847322624	g_fielding	RT @Jisc: Here it is - our list of the UK's to...	2016-03-04 16:53:59	en
			RT @Jisc: Slides	2016-03-	

9	705798322263556097	hammel_rachel	and resources from two very b...	04 16:53:27	en

The code snippet below sets up the displaying of the structure of the data set by getting the necessary information needed to create the pie chart. It also shows the first 10 entries in the dataframe types.

In [3]:

```
types = analyse_types(data)
types.head(10)
```

Out[3]:

	tweet	retweet	reply
68963	1	NaN	NaN
284553	NaN	2	NaN
726453	NaN	1	NaN
740343	NaN	NaN	1
763418	1	NaN	NaN
769897	NaN	1	NaN
1147031	4	NaN	NaN
1168961	2	2	3
1186911	39	5	2
1210901	NaN	1	NaN

In [4]:

```
types.describe()
```

Out[4]:

	tweet	retweet	reply
count	811.0	1844.0	152.0
unique	58.0	48.0	14.0
top	1.0	1.0	1.0
freq	327.0	1152.0	97.0

In [5]:

```
types.mean()
```

Out[5]:

```
tweet      7.643650
retweet    3.204447
reply      2.572368
dtype: float64
```

In [6]:

```
types.std()
```

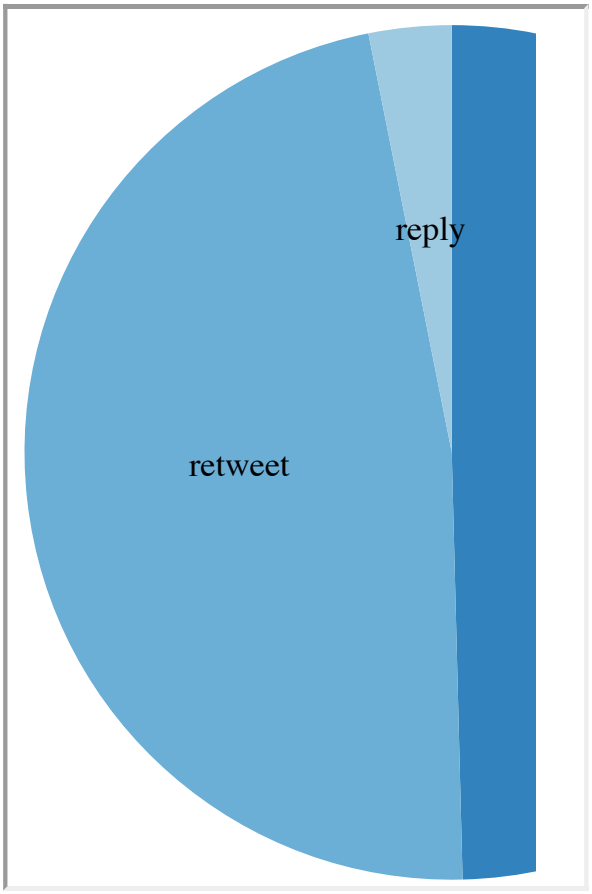
Out[6]:

```
tweet      18.879279
retweet     8.262117
reply       4.872686
dtype: float64
```

The code beneath calls the html file that loads the data that was refined in analyse_types to a pie chart

In [7]:

```
%%html
<iframe width="50%" height ="415" align = "left" src="Display/types_pieChart.h
```



The pie chart above shows the structure of the tweets, retweets replies. From the graph you can see that out of the three types tweet was the most popular followed by retweet and finally reply.

The below code fragment sets up the graph for the timeline of tweets, retweets and replies during the period of the event being looked at. The tweets,retweets and replies are grouped hourly. This is done inorder to get the most readable and accurate data possible to analyse effectively. It also shows what is contained in the dataframe tweet_time.

In [8]:

```
tweet_time = analyse_types_field(data, "created_at", "h")
encode_json(tweet_time, "tweet_time.json")
tweet_time.head(10)
```

Out[8]:

	tweet	retweet	reply
2016-02-22 10:00	3	NaN	NaN
2016-02-22 11:00	2	4	NaN
2016-02-22 12:00	NaN	6	NaN
2016-02-22 13:00	1	5	NaN
2016-02-22 14:00	NaN	3	NaN
2016-02-22 15:00	2	2	NaN
2016-02-22 16:00	1	1	NaN
2016-02-22 17:00	NaN	3	NaN
2016-02-22 18:00	NaN	1	NaN
2016-02-22 20:00	1	1	NaN

In [9]:

```
tweet_time.describe()
```

Out[9]:

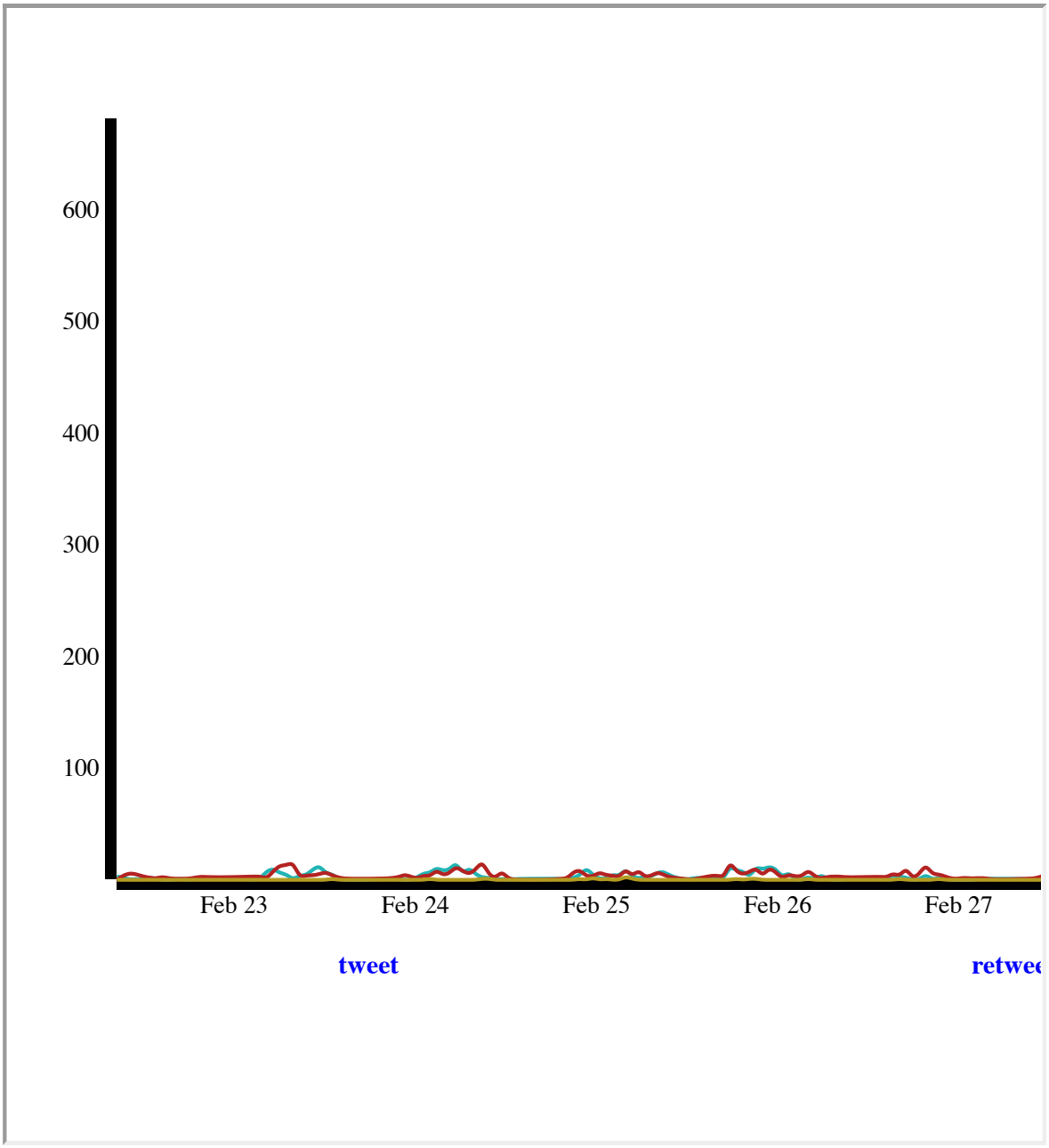
	tweet	retweet	reply
count	182.0	217.0	69
unique	47.0	56.0	17
top	1.0	1.0	1
freq	40.0	31.0	29

The below code snippet sets up the displaying of the timeline graph. Note: This graph is interactive. One can click on the tweet types label inorder to toggle to show the line or not too.

- The green line graph is that of the tweet
- The red line graph is that of the retweet
- The orange line graph is that of the reply

In [10]:

```
%%html
<iframe width="100%" height ="600" src="Display/tweets_per_time.html"></iframe
```



The graph above shows the timeline of the tweets created during the time period of when the hashtag was used. It plots the data hourly. One can see that the weeks leading up to the event that many users were not active in sending tweets. It can be seen that there were fairly the same amount of both retweets and tweets sent during 22nd of Feb till the 29th of Feb. The retweets being sent slightly more than tweets. However during the 2nd of march you can see that more tweets were sent then retweets. One can observe from the graph that during the last few days of the event the activity of sending out different types of tweets have increased exponentially. This is especially apparent during March 01 and March 03. This could be due to the fact that the digifest16 event had reached it's main event and therefore the audience may have had more to say at the time. Another interesting observation that can be made in by looking at the graph is that between the 2nd of March and the 3rd there is a dip in activity. This could possibly be due to the fact that it was during the middle of the night and so very few

users would actually be tweeting. If one looked at the graph closely one could possibly see a pulse. Where there are intervals of activity followed by a lull. This could highly likely be due to the sleep patterns people have.

The python code below creates the data needed in order to make the bar graph possible for the frequency of applications used to send tweets. It also shows the first 10 entries in the data frame apps.

In [11]:

```
apps = analyse_field(data, 'source')
encode_json(apps, "Apps_count.json")
apps.head(10)
```

Out[11]:

	count	source
0	57	Buffer
1	60	DoubleDutch
2	275	GaggleAMP
3	774	Hootsuite
4	67	Mobile Web (M5)
5	101	RoundTeam
6	51	TweetCaster for Android
7	1569	TweetDeck
8	60	Tweetbot for iOS
9	2726	Twitter Web Client

In [12]:

```
apps.describe()
```

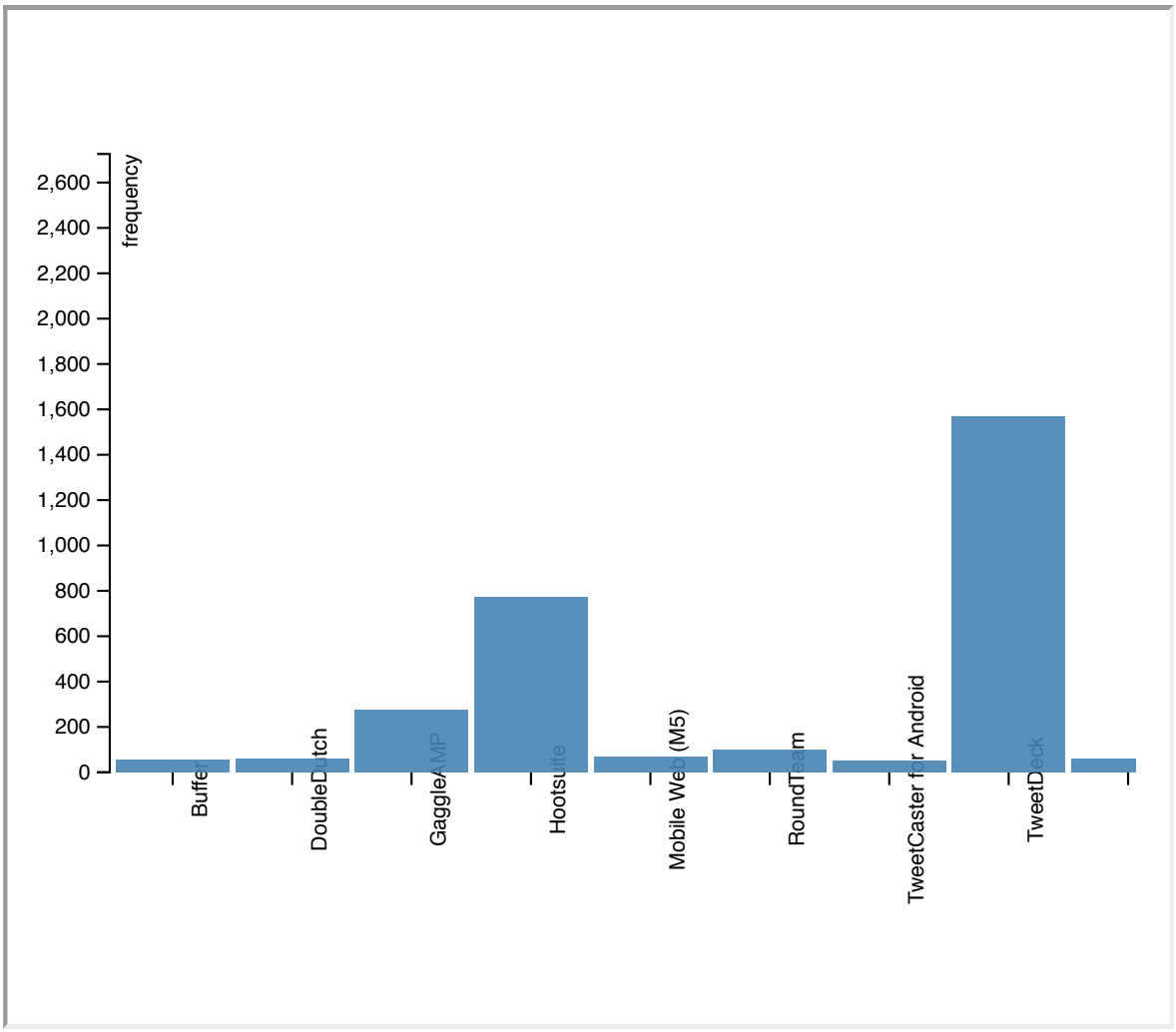
Out[12]:

	count
count	16.00000
mean	781.18750
std	947.99657
min	51.00000
25%	65.25000
50%	319.50000
75%	1346.25000
max	2726.00000

The code snippet below sets up the displaying of the timeline graph. Note: This graph is interactive as you can sort the values from most frequent to least frequent. Also hovering over will highlight the bar one is hovering over.

In [13]:

```
%%html
<iframe width="100%" height ="480" src="Display/appsToSendTweets.html"></ifram
```



The bar graph above shows the applications that were used the most to send tweets out into the world wide web. One can see that Twitter Web Client, Twitter for iPhone and Twitter for Android are the most popular devices to send tweets. This is hardly surprising as they are the most accessible ways of accessing the internet. When the graph is sorted, one can see that there are not many applications that are used at a frequency of 1000. There seems to be only five such sources that are very popular. From this graph one can also make an educated guess as to which application is most popular in the current market.

The python code below creates the data needed in order to make the hashtag cloud possible for the frequency of each Hashtag being tagged. It also shows the first 10 entries in the data frame hashtags.

In [14]:

```
hashtags = analyse_entities(data, 'hashtags', 'text')
encode_json(hashtags, "displayData.json")
hashtags.head(10)
```

Out[14]:

	count	hashtags
0	1	14JCID
1	10	3dprinting
2	2	4life
3	1	AI
4	2	AIDAthatsnotmyname
5	4	AR
6	2	ARUWomensNetwork
7	4	Aberdeen
8	2	Adele
9	3	Africa

In [15]:

```
hashtags.describe()
```

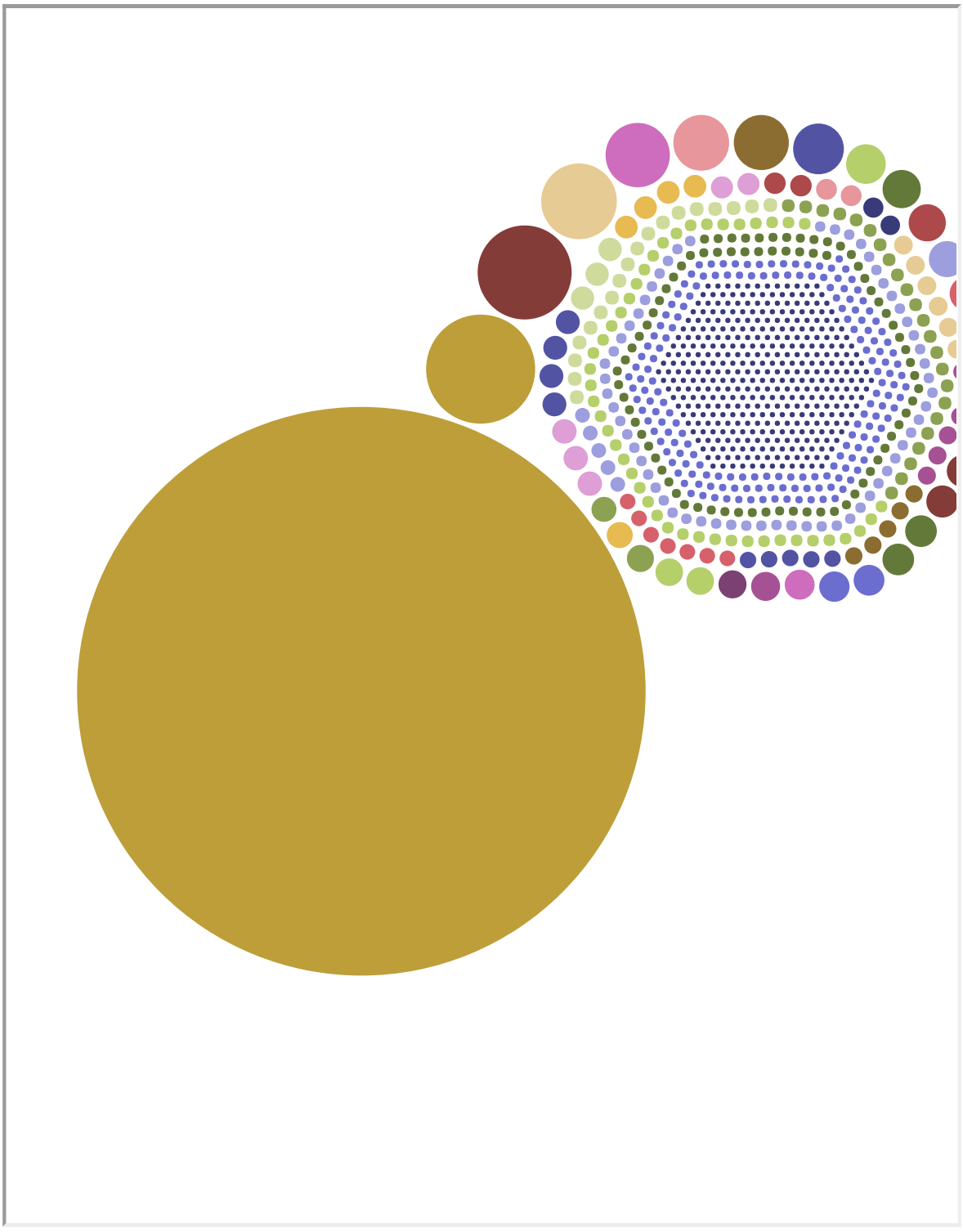
Out[15]:

	count
count	849.000000
mean	19.680801
std	411.060772
min	1.000000
25%	1.000000
50%	2.000000
75%	4.000000
max	11965.000000

The code snippet below sets up the displaying of the hashtag cloud. Note: This graph is interactive as you can hover the clouds and this will highlight the relevant data needed to be shown.

In [17]:

```
%%html
<iframe width="100%" height ="700" src="Display/tweetBubble.html"></iframe>
```



The hashtag "digifest16" is the largest cloud. This is due to the fact this hashtag was used inorder to pull the data. one thing to keep in mind is that this cloud could have been a lot larger then it is if during the analysis the one were to ignore upper cases. This graph is quite useful for telling the most popular trends at the time.

The below function call would have been used to calculate the interactiosn each user had with each other. However there was not enough time to finish the graph for it

In [18]:

```
analyse_interactions(data)
```

The python code below creates the data needed in order to make the hashtag cloud possible for the frequency of each user being mentioned. It also shows the first 10 entries in the data frame mentions.

In [19]:

```
mentions = analyse_entities(data, 'user_mentions', 'id_str')
encode_json(mentions, "mentions.json")
mentions.head(10)
```

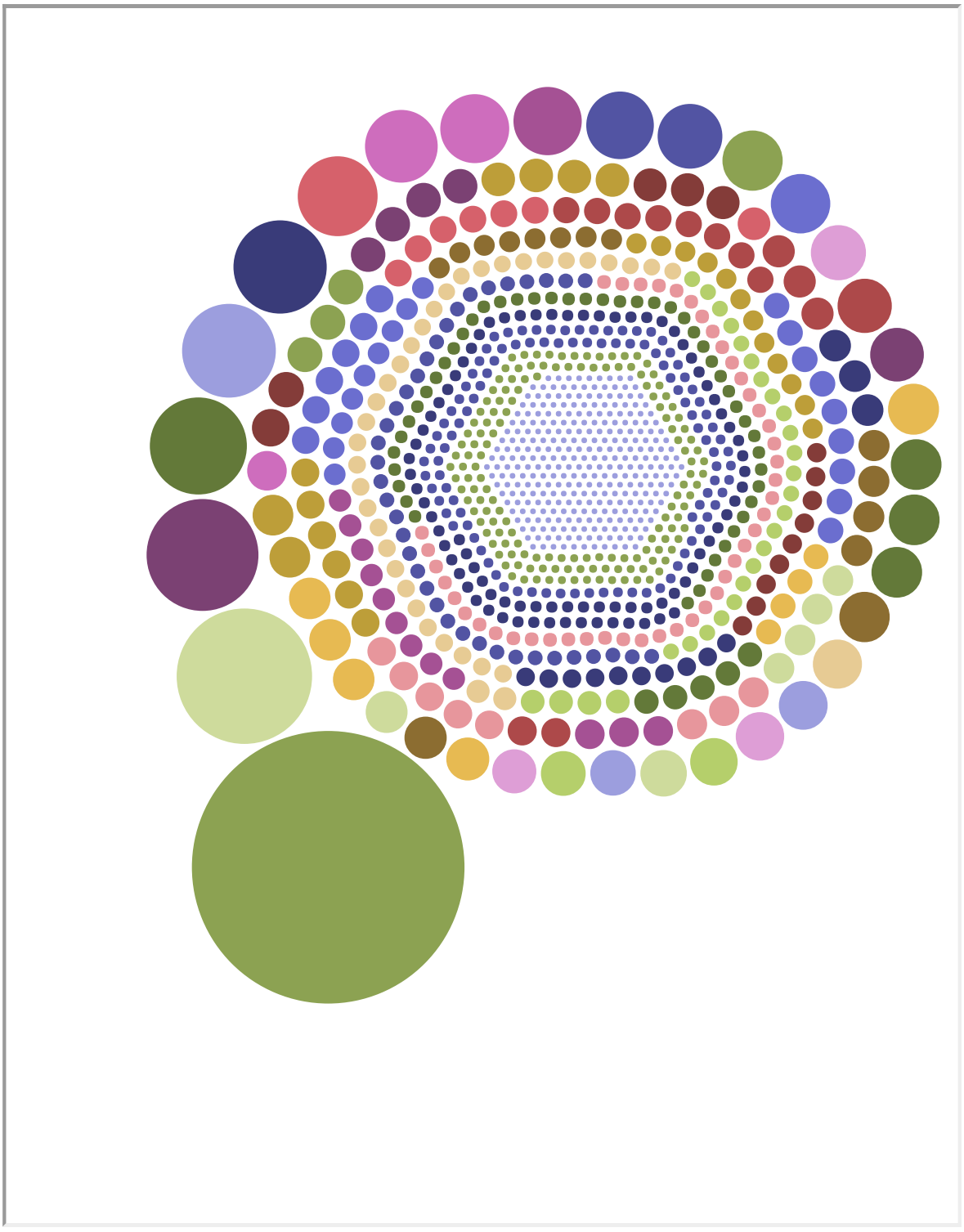
Out[19]:

	count	user_mentions
0	4	100029807
1	3	1002679603
2	4	1019043228
3	4	102077957
4	15	1022476932
5	1	10228272
6	4	102646736
7	5	102654014
8	2	102685431
9	1	102951812

The code snippet below sets up the displaying of the mentions cloud. Note: This graph is interactive as you can hover the clouds and this will highlight the relevant data needed to be shown.

In [20]:

```
%%html
<iframe width="100%" height ="700" src="Display/popularMentions.html"></iframe
```



The above cloud shows the most popular user. This is shown by the number of times the user was mentioned.

The python code below creates the data needed in order to make the replies cloud possible for the frequency of each user being replying. It also shows the first 10 entries in the data frame replies.

In [21]:

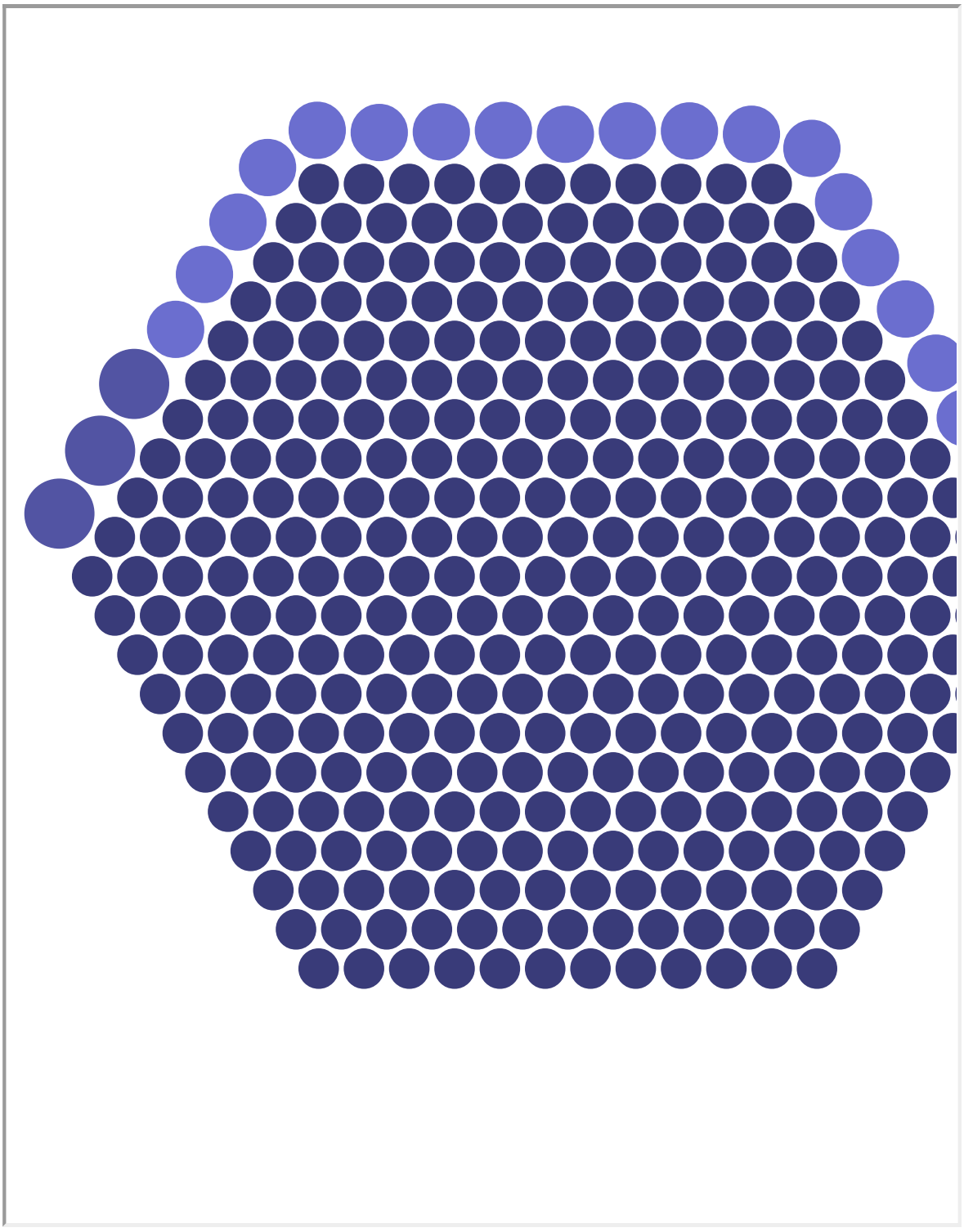
```
replies = analyse_replies(data)
encode_json(replies, "replies_per_user.json")
replies.head(10)
```

Out[21]:

```
6.897321e+17    1
6.917383e+17    3
6.977457e+17    1
7.022103e+17    1
7.024451e+17    1
7.025329e+17    2
7.025612e+17    1
7.025711e+17    1
7.026297e+17    1
7.027760e+17    1
dtype: int64
```

In [22]:

```
%%html
<iframe width="100%" height ="700" src="Display/replyBubble.html"></iframe>
```



The above cloud shows the user that replied the most. This is shown by the number of times the user replied.

The python code below creates the data needed in order to make the retweet cloud possible for the frequency of each user being replying. It also shows the first 10 entries in the data frame retweets.

In [23]:

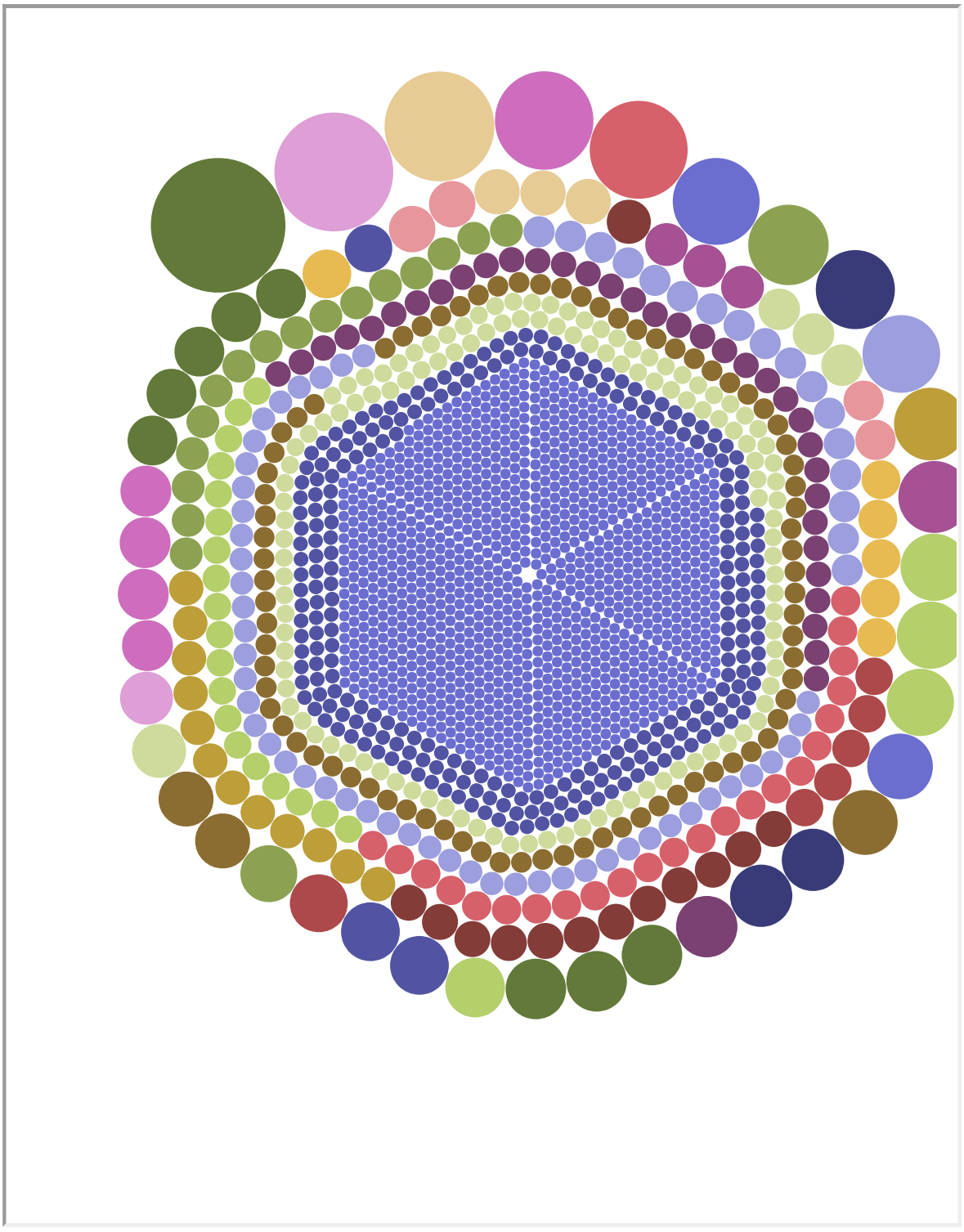
```
retweetsBubble = analyse_retweets(data)
encode_json(retweetsBubble, "retweets_per_user.json")
retweetsBubble.head(10)
```

Out[23]:

	retweet
68963	NaN
284553	2
726453	1
740343	NaN
763418	NaN
769897	1
1147031	NaN
1168961	2
1186911	5
1210901	1

In [24]:

```
%%html
<iframe width="100%" height ="700" src="Display/retweetsBubble.html"></iframe>
```



The above cloud shows the user who was retweeted the most. This is shown by the number of times the was retweeted.